EP 31884 (2)

(12) UK Patent Application (19) GB (11) 2 347 530 (13) A

(43) Date of A Publication 06.09.2000

(21) Application No 9919243.7

(22) Date of Filing 13.08.1999

(30) Priority Data
(31) 09260391 (32) 01.03.1999 (33) US

(51) INT CL^7
G06F 9/44

(52) UK CL (Edition R)
G4A AUXX

(56) Documents Cited
US 5471925 A
Charles Petzold,"Programming Windows", pubd.1988,MicrosoftPress,see esply. pp.9, 10,14-17,25-28,281-321

(58) Field of Search
UK CL (Edition Q) G4A AUXX
INT CL^6 G06F 9/44

(71) Applicant(s)
Mitel, Inc
(Incorporated in USA - Delaware)
205 Van Buren Street, Suite 400, Herndon, Virginia 22070, United States of America

(72) Inventor(s)
Julio Ortiz
Stephen G Marth
David Randall Ronca

(74) Agent and/or Address for Service
Reddie & Grose
16 Theobalds Road, LONDON, WC1X 8PL, United Kingdom

(54) Abstract Title
Branding dynamic link libraries

(57) A data processing system is arranged for use with a computer product such as a software application to display branding information for various hardware or software components being used by the application. In a representative computer product the branding data displayed in a statistics reporting application are represented by the product name 12, the company name 14, and the company logo 16.

The branding information associated with the computer product are stored in a central library, and a routine to access the branding information is stored in a second library. The accessing routine is called by a software application requesting branding data, and the appropriate branding data are extracted from the central library in response to the call. The central and second libraries may be dynamic link libraries.

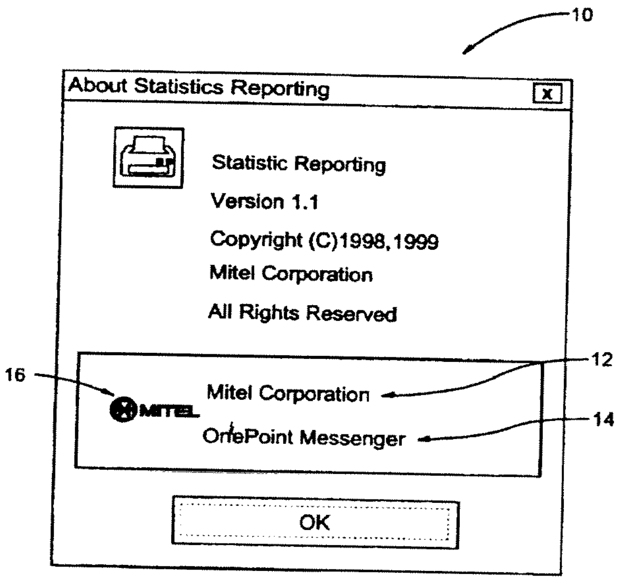
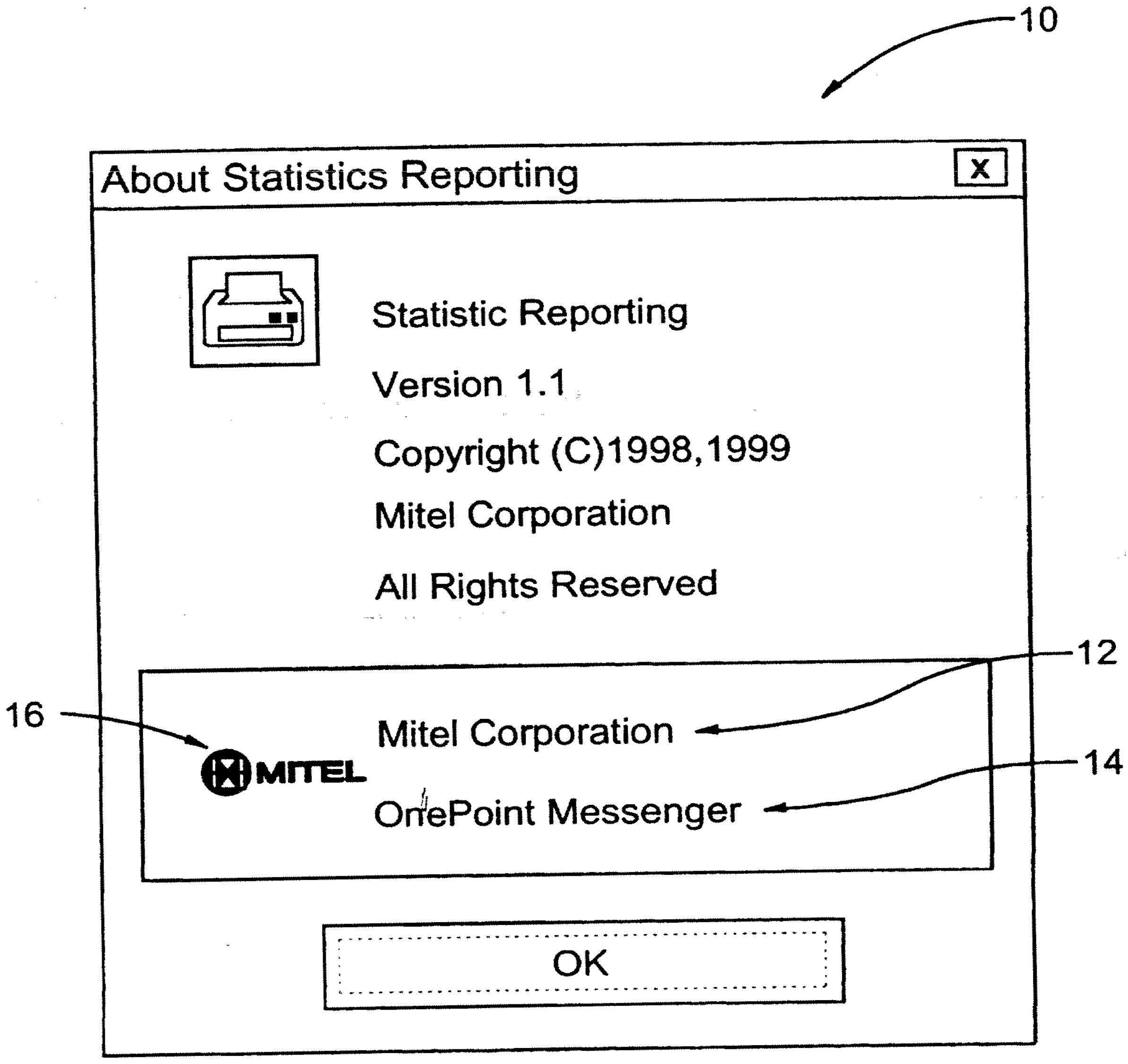


FIG.1

GB 2 347 530 A

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.



Statistic Reporting

Version 1.1

Copyright (C)1998,1999

Mitel Corporation

All Rights Reserved

Mitel Corporation

OnePoint Messenger

FIG.1

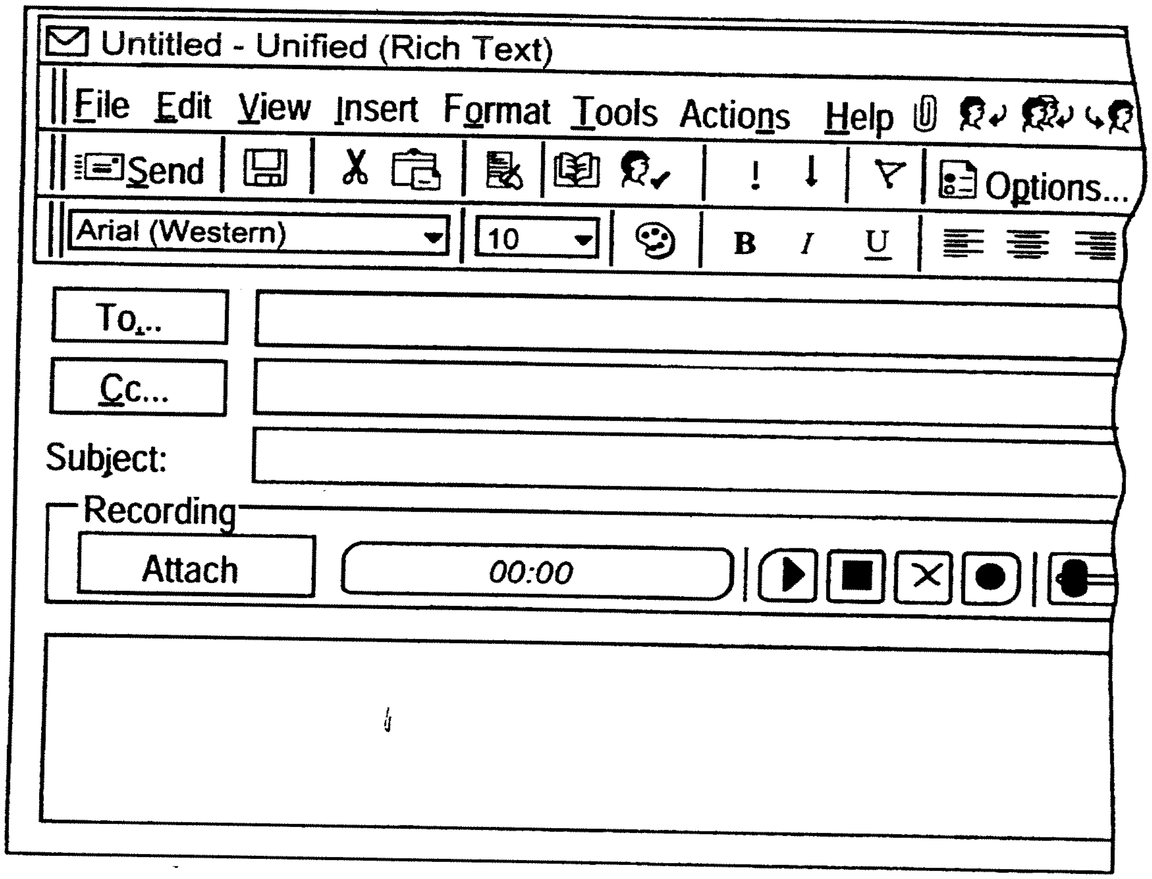

Untitled - Unified (Rich Text)
File Edit View Insert Format Tools Actions Help
Send
Options...
Arial (Western)
10
B I U
To...
Cc...
Subject:
Recording
Attach
00:00

FIG.2A

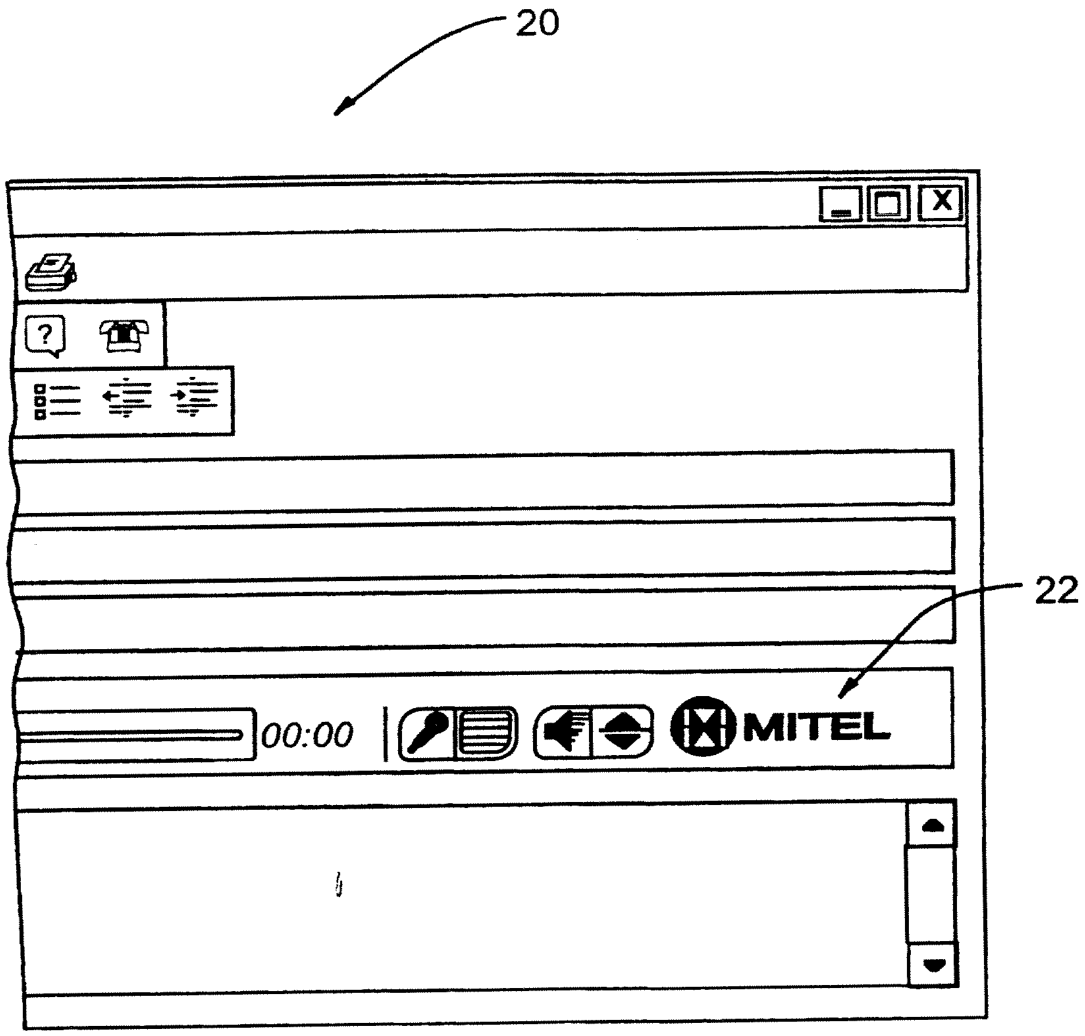


FIG.2B

2347530

BRANDING DYNAMIC LINK LIBRARIES

Field of the Invention

The present invention relates in general to product marking and, more particularly, providing branding data to applications for presentation on a display and/or graphical user interface.

Background of the Invention

Marking products with company logos and/or trademarks is common practice. In the computer industry, hardware and software components used to form a computer product often have associated trademarks and are often supplied by a number of different companies. These hardware and software components for the most part remain unseen to end users. Notwithstanding this, it is still desired to present company logos and trademarks associated with the hardware and software components. To achieve this, branding data providing product and/or company names is sometimes included with the hardware and software components. The branding data associated with each hardware and software component is typically stored in a separate file in memory. Unfortunately, since the branding data for the various hardware and software components is stored in separate files, the branding data is time consuming to access and difficult to update especially when it is necessary to access branding data associated with multiple components.

Thus there is a need for improved techniques to manage branding data in a computer product.

Summary of the Invention

The present invention provides for centralized management of branding information for a computer product. The present invention is particularly well suited for use with software application to display branding information for various hardware and/or software components being utilized by the software application.

According to one aspect of the present invention there is provided a method for managing branding data within a computer product. The method includes the operations of: storing branding information associated with the computer product in a central library; and storing at least one routine to access the branding information in at least one second library, the at least one routine being called by a software application requesting branding data and extracting the appropriate branding data from the central library in response to the call.

According to another aspect of the present invention there is provided a method of accessing branding data stored within a computer product in response to a request from a software application. The method includes the operations of: linking to a first library storing routines to access branding data stored in a central library in response to the request; calling the routines in the first library, the called routines loading the central library and extracting branding data from the central library identified in the request; and conveying the extracted branding data to the software application.

In one embodiment, the central and first libraries are dynamic link libraries. The branding data can include product names, original equipment manufacturer (OEM) names and images. The branding data can be stored in a version resource in the central library and can include string resources for the product names and OEM names. The images can also be stored in the version resource in bitmap resources.

According to another aspect of the present invention there is provided a computer readable medium including computer program code for accessing branding data stored in a central resource. The computer readable medium includes: computer program code for receiving a branding data request from a software application; computer program code for assessing the central resource to retrieve pertinent branding data based on the branding data request; and computer program code for conveying the pertinent branding data to the software application in response to the branding data request.

The present invention provides advantages in that the branding data is kept in a compact form, which can be easily accessed and updated. As a result, it is a simple and direct procedure to add new procedures and branding data and to alter existing procedures and branding data.

Brief Description of the Drawings

An embodiment of the present invention will now be described more fully with reference to the accompanying drawing in which:

Figures 1 and 2 illustrate displaying branding information within representative windows (forms) of a computer product.

Detailed Description of the Invention

As mentioned previously, it is often desired to present company logos and trademarks associated with hardware and software components embodied in a computer product on a display and/or graphical user interface. Computer product within the context of this application refers to virtually any product including a processing unit executing software code.

The present invention provides for centralized management of branding information for a computer product. The present invention is particularly well suited for use with software application to display branding information for various hardware and/or software components being utilized by the software application. Figures 1 and 2 illustrate displaying branding information within representative windows (forms) of a computer product (e.g., software application).

Figure 1 represents a screen shot 10 of an About screen from a representative computer product. Here, the computer product is a software application pertaining to statistics reporting know as "OnePoint Messenger" by Mitel Corporation. Here, "One point Messenger" is a product name 12, and Mitel Corporation is a company name 14 which produced the product. The company name 14 can refer to an Original Equipment Manufacturer (OEM) for the product (e.g., Mitel Corporation). The company logo 16 for Mitel Corporation is also provided. Thus, the About screen provides the important branding information for the computer product. The product name 12, the company name 14 and the company logo 16 represent the branding information being displayed.

Figure 2 is a screen shot 20 of a application window for a compose form for composing unified messages in a messaging system. Besides the typical areas for buttons and text entry fields for a compose form, the screen shot also illustrates a company logo 22 on the compose form. The company logo 22 represents the branding information being displayed in this screen shot 20. Thus, branding information can be provided on useful areas of the computer product (e.g., software applications) which are typically displayed to users.

As will be appreciated, hardware and software components within the computer product may have trademarks associated with them and/or may be produced by different companies. The different companies can pertain to the various hardware and software components. Often, many of the different companies are Original Equipment Manufacturers (OEMs). As a result these hardware and software components may have associated branding data.

To facilitate access to branding data, the present invention provides software to manage branding data in a computer product. The software makes using of dynamic link libraries (DLLs), which provide a simple and compact procedure for software applications to access required branding data. Specifics of the branding data software will now be described.

In order to manage branding data, two DLLs are created, namely cvBrand.DLL and cvOEMBrand.DLL. cvBrandDLL holds a number of routines which are called in response to a request for branding data made by a software application to access branding data. cvOEMBrand.DLL holds the actual branding data. Thus, all branding data is stored in a single central location. In the present embodiment, the cvOEMBrand.DLL includes a version resource with string resources in English UNICODE. The string resources include product name strings, OEM name strings and bitmaps of branding images.

In the present embodiment, the cvBrand.DLL holds four routines, namely a BOOL getProductName routine, a BOOL getOEMName routine, a BOOL getImage routine and a BOOL getName routine. Appendix A illustrates the variables associated with these routines. These four routines when called in response to a request for branding data allow product name, OEM name and image branding data to be loaded from the cvOEMBrand.DLL. When the branding data is loaded by cvBrand.DLL, the branding data can be accessed by the requesting software application. As will be appreciated, since the branding data is accessed by making DLL function calls, the branding data retrieval process is simple and can be carried out quickly.

In operation, when a software application requires branding data, the software application dynamically or statically links to the cvBrand.DLL and calls the routines of the cvBrand.DLL to access the appropriate product name, OEM name and product image branding data from the cvOEMBrand.DLL. Once the cvOEMBrand.DLL is accessed, the branding data is extracted by the cvBrand.DLL and is conveyed to the software application.

When the BOOL getProductName routine is called, the routine makes a getName call with the variable ntName set to ID_PRODUCT identifying the product name for which branding data is required. This call is used by the BOOL getName routine as will be described.

When the BOOL getOEMName routine is called, the routine also makes a getName call with the variable ntName set to ID_OEM identifying that OEM branding data is required. This call is also used by the BOOL getName routine as will be described.

When the BOOL getImage routine is called, a standard LoadLibrary call is made to load the cvOEMBrand.DLL simply using an unpathed filename. If the cvOEMBrand.DLL is not loaded in response to the call, the getImage routine identifies the correct path by using a GetModuleFileName call to identify where the cvOEMBrand.DLL is located. The getImage routine replaces the cvOEMBrand.DLL name with the pathed name OEM_DLL_FNAME, which identifies the OEM DLL filename and tries to load the cvOEMBrand.DLL using the pathed name. If the cvOEMBrand.DLL is still not loaded, a message box announces an error message and the getImage routine returns a value of FALSE.

If the LoadLibrary call does not return a value of FALSE, the getImage routine next calls a second standard LoadImage routine to load a requested logo, as determined by the iiImageID variable from the cvOEMBrand.DLL. A standard FreeLibrary routine then releases the cvOEMBrand.DLL. If the LoadImage routine does not return a value of FALSE, the "phIMageHandle" variable is set to point to the value returned by the LoadImage routine.

When getProductName and getOEMName routines make getName calls, the BOOL getName routine is called. When the getName routine is called, the getName routine firstly checks if wLangID = 0 and if so, sets wLangID to the systems default Language ID; otherwise it uses the given value. Next, a request to the cvOEMBrand.DLL version resource is prepared.

If ntName is ID_PRODUCT and the buffer_size variable is large enough to hold the maximum allowed product name (as determined by variable MAX_PRODUCT_NAME_LENGTH), then a request string cTmpStr is written to find a ProductName string (using wLangID). In the preferred embodiment, the buffer_size variable is set to 32 characters. If the request string is unable to find the ProductName string, the request string is set to point to a default failure string MISSING_PRODUCT_NAME string , currently set to "[]".

If ntName is ID_OEM and the buffer_size variable is large enough to hold the maximum allowed OEM name (as determined by variable MAX_OEM_NAME_LENGTH), then a request string cTmpStr is written to find an OEMName string (using wLangID). In the preferred embodiment, the buffer_size variable is set to 32 characters. If the request string is unable to find the OEMName string, the request string is set to point to a default failure string MISSING_OEM_NAME string , currently set to "[]".

The getName routine then reads the appropriate string from the cvOEMBrand.DLL version resource. During the read, the getName routine retrieves information such as the version information size (variable dwVerSize) using a GetFileVersionInfoSize routine stored in a standard version.lib library, extracts a copy of the version resource into an lpData variable using a GetFileVersionInfo routine from the version.lib library, and uses the cTmpStr request string to copy a correct string value into the lpData variable via a VerQueryValue routine from version.lib library. If the string is correctly extracted, the string is copied into the buffer variable.

The getName routine returns a value of TRUE unless ntName is neither ID_PRODUCT or ID_OEM or if the buffer_size variable is not large enough to hold the entire string name.

The invention can also be embodied as computer readable code on a computer readable medium. The computer readable medium is any data storage device that can store data which can be thereafter be read by a computer system. Examples of the computer readable medium include read-only memory, randomaccess memory, CD-ROMs, magnetic tape, optical data storage devices. The computer readable medium can also be distributed over a network coupled computer systems so that the computer readable code is stored and executed in a distributed fashion.

Although a preferred embodiment of the present invention has been described, those skilled in the art will appreciate that variations and modifications may be made without departing from the spirit and scope thereof as defined by the appended claims.

APPENDIX A

BOOL getProductName(WCHAR * buffer, short buffer_size, WORD wLangID)

where:

variable "buffer" is used to hold a product name string;

variable "buffer_size" lists the largest number of wide characters to be copied (presumably the same number as the size of the "buffer" variable); and

variable wLangID" holds the binary language-id/code-page pair. In the preferred embodiment, wLangID is not implemented and is therefore set to 0

BOOL getOEMName(WCHAR * buffer, short buffer_size, WORD wLangID)

where:

variable "buffer" is used to hold the OEM name string;

variable "buffer_size" lists the largest number of wide characters to be copied (presumably the same number as the size of the "buffer" string); and variable wLangID" holds the binary language-id/code-page pair. In the preferred embodiment, wLangID is not implements and is therefore set to 0.

BOOL getImage (short iiImageID, HANDLE * phIMageHandle)

where:

variable iiImageID is of type IMAGE_ID and takes a value of either SMALL_LOGO or LARGE_LOGO; and

variable phImageHanldle is a handle pointer that holds an image handle.

BOOL getName(WCHAR * buffer, short buffer_size, WORD wLangID, NAME_TYPE ntName)

where:

the variables "buffer", "buffer_size" and "wLangID" are the same as those called in the first two routines.

We Claim:

1. A method for managing branding data within a computer product comprising:

storing branding information associated with said computer product in a central library; and

storing at least one routine to access said branding information in at least one second library, said at least one routine being called by a software application requesting branding data and extracting the appropriate branding data from said central library in response to said call.

2. A method of claim 1 wherein said at least one second library stores a plurality of routines, each of said routines being called to extract different types of branding data from said central library.

3. A method of claim 2 wherein said routines are stored in a single second library.

4. A method of claim 3 wherein said central and second libraries are dynamic link libraries.

5. A method of claim 4 wherein said software application requesting branding data statically or dynamically links to said second library and calls the routines therein to access said branding data.

6. A method of claim 4 wherein said different types of branding data include product names, original equipment manufacturer (OEM) names and images.

7. A method of claim 6 wherein said branding data is stored in a version resource including string resources for said product names and OEM names.

8. A method of claim 7 wherein said images are stored in said version resource in bitmap resources.

9. A method of accessing branding data stored within a computer product in response to a request from a software application comprising:

linking to a first library storing routines to access branding data stored in a central library in response to said request;

calling the routines in said first library, said called routines loading said central library and extracting branding data from said central library identified in said request; and

conveying the extracted branding data to said software application.

10. A method of claim 9 wherein said central and first libraries are dynamic link libraries.

11. A method of claim 10 wherein said software application requesting branding data statically or dynamically links to said first library.

12. A method of claim 11 wherein said branding data include product names, original equipment manufacturer (OEM) names, and images.

13. A method of claim 12 wherein said branding data is stored in a version resource in said central library and includes string resources for said product names and OEM names.

14. A method of claim 13 wherein said images are stored in said version resource in bitmap resources.

15. A computer readable medium including computer program code for accessing branding data stored in a central resource, said computer readable medium comprising:

The Patent Office

INVESTOR IN PEOPLE

12

Application No:	GB 9919243.7	**Examiner:**	Leslie Middleton
Claims searched:	1,9,15	**Date of search:**	2 September 1999

Patents Act 1977
Search Report under Section 17

Databases searched:

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.Q): G4A (AUXX)

Int Cl (Ed.6): G06F 9/44

Other: Online: EPODOC, PAJ, WPI / EPOQUE

Documents considered to be relevant:

Category	Identity of document and relevant passage	Relevant to claims
X	US5471925 A (Francotyn-Postalia) See columns 4-6	1,9,15 at least
X	Charles Petzold, "Programming Windows", pubd. 1988, Microsoft Press, see especially pages 9,10,14-17,25-28,281-321.	1,9,15 at least

X	Document indicating lack of novelty or inventive step	A	Document indicating technological background and/or state of the art.
Y	Document indicating lack of inventive step if combined with one or more other documents of same category.	P	Document published on or after the declared priority date but before the filing date of this invention.
&	Member of the same patent family	E	Patent document published on or after, but with priority date earlier than, the filing date of this application.

computer program code for receiving a branding data request from a software application;

computer program code for assessing the central resource to retrieve pertinent branding data based on the branding data request; and

computer program code for conveying the pertinent branding data to the software application in response to the branding data request.

16. A computer readable medium as recited in claim 15 wherein said computer readable medium further comprises:

computer program code for displaying the pertinent branding data to a user of the software application.

17. A computer readable medium as recited in claim 15 wherein said computer program code for accessing uses at least one DLL.

18. A computer readable medium as recited in claim 15 wherein the central resource is provided as a DLL.